человек» понял и признал, что дар зрения чреват, двусмыслен, опасен – и неизбежен.

1. *Konersmann R.* Kritik des Sehens. Leipzig, 1997.

2. Modernity and the Hegemony of Vision / Ed. by D. Michael. Berkeley, 1993.

3. Sites of Vision. The Discursive Construction of Sight in the History of Philosophy / Ed. by D. Michael. Cambridge (Mass.); L., 1997.

4. The Opening of Vision: Nihilism and the Postmodern Situation / Ed. by D. Michael. N. Y.; L., 1988.

5. Michael D. The Philosopher's Gaze. Berkeley, 1999.

С. С. Кралин

АРТИКУЛЯЦИЯ БЫТИЯ В ЯЗЫКЕ

Под «языком» в данной статье мы будем понимать «язык программирования». Попытаемся ассоциировать с концептом «бытие» – в качестве если не примера, то функционального коррелята[1] – некоторую вполне эмпирическую реальность.

Назовем бытием объекта область памяти, используемую для его хранения. Тогда результаты выполнения некоторого кода могут быть сопоставлены с известными из истории философии высказываниями о бытии. При этом окажется отраженным многообразие этих последних – с учетом принятия или непринятия в том или ином языке требований *a*) явной инициализации и *b*) строгой типизации.

Рассмотрим возможные реакции транслятора или среды исполнения на следующий фрагмент псевдокода.

Листинг 1

```
10 new Integer num; // «создание» целого числа num
11 new Boolean val; // «создание» булевой величины val
12 print(num==val); // сообщение о равенстве num и val
```

[1] Поскольку наши рассуждения имеют предметом некоторый *дискурс*, постольку – хотя бы из соображений этического толка – мы и говорим о них как о находящихся с ним в отношении «функциональной корреляции», а не в отношении «концепт – пример». Удачным словосочетанием «функциональный коррелят» мы обязаны Н. В. Бряник.

Ситуация ($a \vee b$) чревата появлением сообщений об ошибках. Требование a проявится чем-нибудь вроде variable is not initialized. Это будет означать, что бытие – это всего лишь бытие сущего, что никакого бытия нет, а есть лишь сущее. Требование b скажется чем-то наподобие types mismatch. Этим будут зафиксированы множественность бытия, наличие многих его «типов»; бытие окажется всегда «своим бытием».

Ситуация ($\sim a \,\&\, \sim b$) позволяет нам избежать появления сообщений об ошибках и получить true при выполнении[2]. Добавим в листинг одну строку

```
13 print(num); // вывод значения num
```

и рассмотрим, что это добавит к имеющемуся выводу.

Во-первых, можно ожидать, что в выводе появится попросту 0. Для нас это будет значить: бытие есть попросту некий особый род сущего, бытие есть сущее среди прочего сущего. Во-вторых, можно надеяться на появление в выводе чего-то сродни null. Это послужит подтверждением того, что бытие в определенном смысле есть то же самое, что и ничто.

Механизмы выделения памяти для объектов, традиционно не являющихся объектами первого класса, довольно специфичны. Представляется, что эта специфика может совпадать со спецификой того, что в истории философии именовалось «человеческим бытием».

Описанные нами корреляции предоставляют семантику для «машинного мышления» о бытии.

---

1. *Хайдеггер М.* Бытие и время. СПб., 2002.

---

[2] Появление false также не исключено. Это произойдет в случае, если переменная автоматически инициализируется «случайным» «мусором» из памяти, или чем-то эквивалентным своему имени, или же адресом того «места» в памяти, где она будет храниться, и т. п. Предлагаем читателю самостоятельно отыскать (по возможности, в [1]) соответствующие этим случаям «положения о бытии».